



**UNIVERSITATEA
TEHNICĂ
DIN CLUJ-NAPOCA**

Restaurant Reservation ChatBot

Limbaje Formale si Translatoare

Autori: Pescaru Silviu, Vlase Rafaella

Grupa: 30235

FACULTATEA DE AUTOMATICA
SI CALCULATOARE

27 Mai 2024

Cuprins

1	Enuntul problemei	2
2	Limbaje, tehnologii si instrumente utilizate	2
3	Descrierea aplicatiei	2
4	Implementare	3
4.1	Code Snippets	3
4.2	Complexitate	4
4.3	Scenariu de test	4
4.4	Avantaje, dezavantaje	6
4.4.1	Avantaje ale utilizarii Azure CLU	6
4.4.2	Dezavantaje	6
4.5	Imbunatatiri ulterioare	6
4.6	Autoevaluare	6
4.7	Link catre aplicatie	6

1 Enuntul problemei

Dezvoltati un chatbot folosind Azure si C# pentru a face o rezervare la un restaurant. Prin intermediul acestui proiect utilizatorul va putea face rezervare la restaurantul dorit, la data si ora alese, poate anula o rezervare, poate verifica rezervarea facuta.

2 Limbaje, tehnologii si instrumente utilizate

- C#
- Azure CLU

3 Descrierea aplicatiei

Discutia cu bot-ul se va realiza in environment-ul Bot Framework Emulator de la Microsoft. Pentru a face posibila comunicarea am avut nevoie de CLU (Conversational Language Understanding), in care am creat noi un Knowledge Base.

Pentru ca modelul creat de noi sa inteleaga propozitiile, am adaugat intentii si entitati. Modelul detecteaza si invata intentiile si extrage entitati din propozitiile pe care le analizeaza.

Iata o lista cu intentiile generale ale unui user:

<input type="radio"/> Intents ↑ ▾	Labeled utteran... ▾	Entities used with this intent ▾
<input type="radio"/> BookTable	27	location, day, hour
<input type="radio"/> Cancel	15	
<input type="radio"/> Confirm	15	
<input type="radio"/> None	0	

Figura 1: Intentii

Am antrenat modelul sa inteleaga diferite entitati din propozitii, cum ar fi locatia, ziua si data rezervarii. Iata un exemplu de antrenament:

<input type="radio"/> BookTable	Is there a spot for a party of seven at <u>Mediterranean Breeze</u> <u>this Friday</u> (26.05.2024) at 20:45?	<u>location</u> <u>day</u> <u>day</u> <u>hour</u>
<input type="radio"/> BookTable	Book a table for two at <u>Green Leaf Café</u> for lunch on <u>Wednesday</u> (31.05.2024) at 13:30.	<u>location</u> <u>day</u> <u>day</u> <u>hour</u>
<input type="radio"/> BookTable	Make a reservation for a group of four at <u>Fusion Flavors</u> on <u>Thursday</u> (26.05.2024) at 19:00.	<u>location</u> <u>day</u> <u>day</u> <u>hour</u>

Figura 2: Entitati

4 Implementare

Partea de backend a fost realizata in C#, plecand de la un sample, CoreBotCLU, existent deja de la Microsoft, ca punct de plecare pentru dezvoltarea chatbot-ului (link: [CoreBotWithCLU](#))

4.1 Code Snippets

In aceasta sectiune vom prezenta cateva portiuni mai semnificative din cod, pentru o mai buna intelegere a procesului de implementare.

```
1 public enum Intent
2 {
3     BookTable,
4     Cancel,
5     Confirm,
6     None
7 }
```

Aici se pot observa intentiile pe care modelul le va invata.

```
1 public class CluEntities
2 {
3     public CluEntity[] Entities;
4
5     public CluEntity[] GetLocationList() => Entities.Where(e => e.Category
6         == "location").ToArray();
7
8     public CluEntity[] GetDayList() => Entities.Where(e => e.Category == "day").ToArray();
9
10    public CluEntity[] GetHourList() => Entities.Where(e => e.Category == "hour").ToArray();
11 }
```

In clasa CluEntities se definesc entitatile generale precum locatia, ziua si ora.

```
1 private async Task<DialogTurnResult> LocationStepAsync(WaterfallStepContext stepContext,
2     CancellationToken cancellationToken)
3     {
4         var bookingDetails = (BookingDetails)stepContext.Options;
5
6         if (bookingDetails.Location == null)
7         {
8             var promptMessage = MessageFactory.Text(LocationStepMsgText,
9                 LocationStepMsgText,
10                 InputHints.ExpectingInput);
11             return await stepContext.PromptAsync(nameof(TextPrompt), new PromptOptions
12                 { Prompt = promptMessage }, cancellationToken);
13         }
14
15         return await stepContext.NextAsync(bookingDetails.Location, cancellationToken);
16     }
```

Acesta este un exemplu de metoda prin care bot-ul preia si proceseaza datele despre locatia data de user.

4.2 Complexitate

Aplicatia are la baza un bot sample de la Azure, care este un nucleu de inceput pentru orice dialog bot. Ca aditie la acest CoreBot am adus partea de dialog specializata pentru rezervari, astfel incat intrebarile puse de bot sa fie relevante in proces. Nu este un proiect foarte complex, scopul sau fiind buna intelegere a crearii unui bot care sa fie capabil sa distinga anumite parti dintr-o propozitie.

4.3 Scenariu de test

In imaginile de mai jos vor fi prezentate exemple de rulare a aplicatiei si exemple de dialog cu Restaurant Reservation Bot

```
PS D:\Facultate\RestaurantBookingBot> dotnet run
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:3978
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: D:\Facultate\RestaurantBookingBot
```

Figura 3: How to run

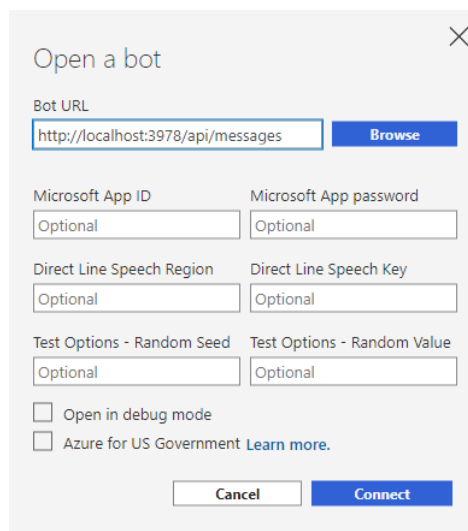


Figura 4: How to open a bot

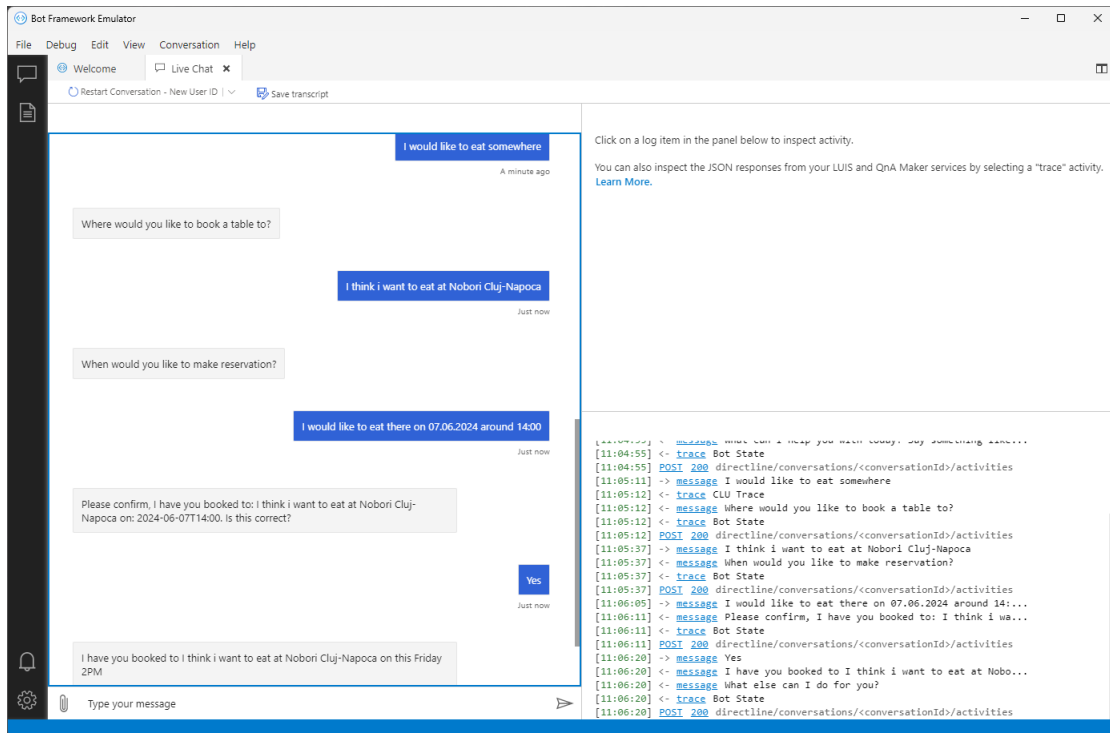


Figura 5: ChatBot dialog

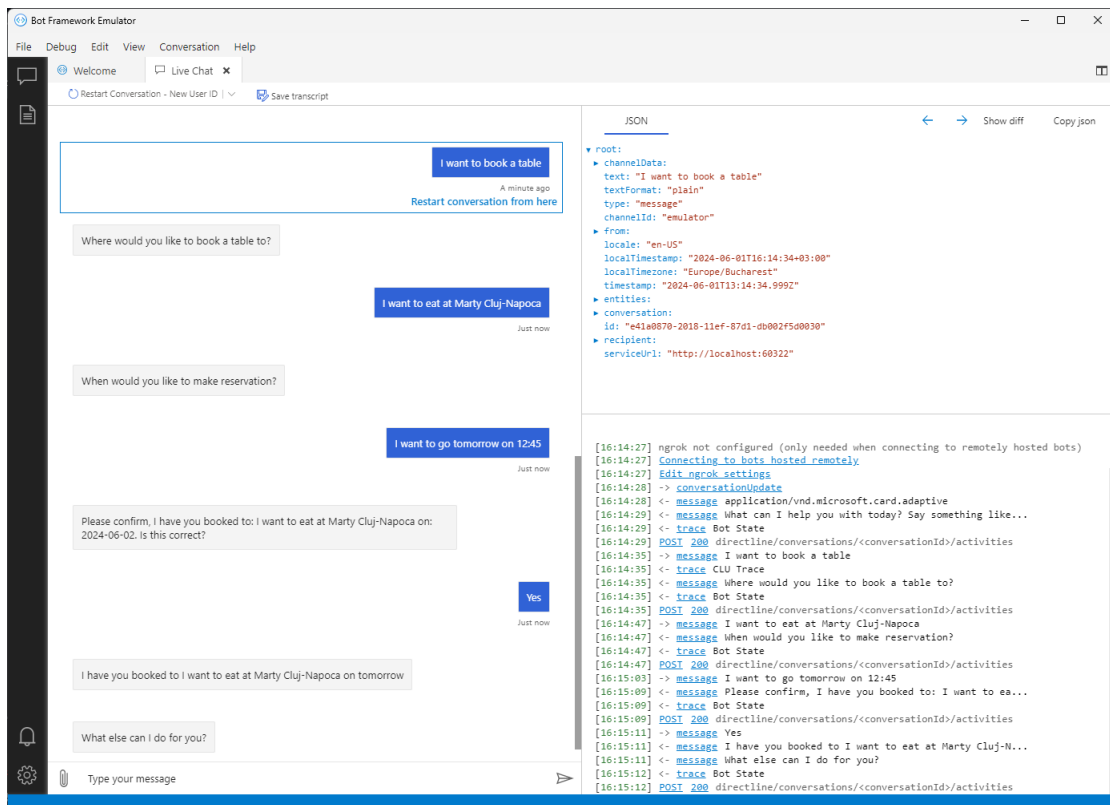


Figura 6: ChatBot dialog

4.4 Avantaje, dezavantaje

4.4.1 Avantaje ale utilizarii Azure CLU

- Antrenarea unui model propriu cu propozitii astfel incat sa nu se hardcodeze raspunsurile primite de la bot
- Aduce un nivel de inteligenta aplicatiei, request-urile se pot formula in mai multe moduri si bot-ul tot va intelege entitatile dorite din propozitie

4.4.2 Dezavantaje

- Pentru a sublinia toate intentiile si entitatile in propozitii este nevoie de timp, plus ca modelul trebuie antrenat pe multe propozitii pentru o acuratete mai mare

4.5 Imbunatatiri ulterioare

ChatBot-ul se poate bucura de o serie de imbunatatiri ulterioare, printre care amintim:

- Utilizare Google API pentru a adauga rezervarile direct in calendarul personal
- Deploy pentru a face bot-ul disponibil pe platforme precum Microsoft Teams sau Whatsapp

4.6 Autoevaluare

Pescaru Silviu: Acest proiect a fost ceva total diferit fata de ce am avut de facut pana acum. A fost prima data cand am lucrat cu Microsoft Azure si toata paleta de Language Studio. La inceput a fost dificil dar dupa ce am inteles lucrurile de baza cum ar fi: definirea intenturilor, entitatilor, cum antrenam un model si cum il conectam la CoreBot prin API; a fost destul de ok de inteles. Un ajutor foarte mare a fost Azure-sample CoreBot care m-a ajutat sa merg repejor cu proiectul.

Vlase Rafaella: Probabil cea mai complicata parte a fost sa inteleg cum sa folosesc Azure pentru a crea un Language Model propriu. Sample-ul CoreBot a fost de ajutor deoarece am avut un schelet al aplicatiei deja implementat. Acest proiect m-a ajutat sa inteleg cum sa antrenez un model sa inteleaga anumite intentii in limbaj natural.

4.7 Link catre aplicatie

- Link catre CoreBot Sample : [CoreBot Sample](#)
- Link catre arhiva Drive a proiectului: [Project Drive Archive](#)